2

DATASTREAM

Request Packet from VT Server to host system

005000000000000000054376200033111000001111000000010000Y

Response Packet from host system to VT Server

005800000000000000054376203311100000111100+000075000+0000725

Fig. 1

APPLICATION PROGRAMMING INTERFACE

4

6

Request Packet

Field Description	Format	Content
Transaction Code	3 N	033
Credit Union Access Code	3 N	Code associated with each credit union. Assigned by host.
Member Number to Withdrawal Funds From	9 N	(entered by caller)
Account Suffix of Withdrawal Funds From	3 N	(entered by caller)
Transfer Amount	9 N	(two decimal positions assumed)
Post Indicator	1 A	N - Preliminary edit; do not update files
		Y - Member has confirmed they want to post this transaction; update files

8

Response Packet

Field Description	Format	Content
Transaction Code	3 N	033
Credit Union Access Code	3 N	
Member Number	9 N	
Host Response Code	3 N	000 - Positive response, continue script
		210-214 - Read error, repeat menu
		220-221 - Read error, repeat menu
Sign field	1 A	+ or -, negative or positive balance
Current Balance of Withdrawal From Account (before transfer)	9 N	(two decimal positions assumed)
Sign field	1 A	+ or -, negative or positive balance
Available Balance of Withdrawal From Account (before transfer)	9 N	(two decimal positions assumed)

Fig. 2

MAPPING DOCUMENT

9

Request Packet

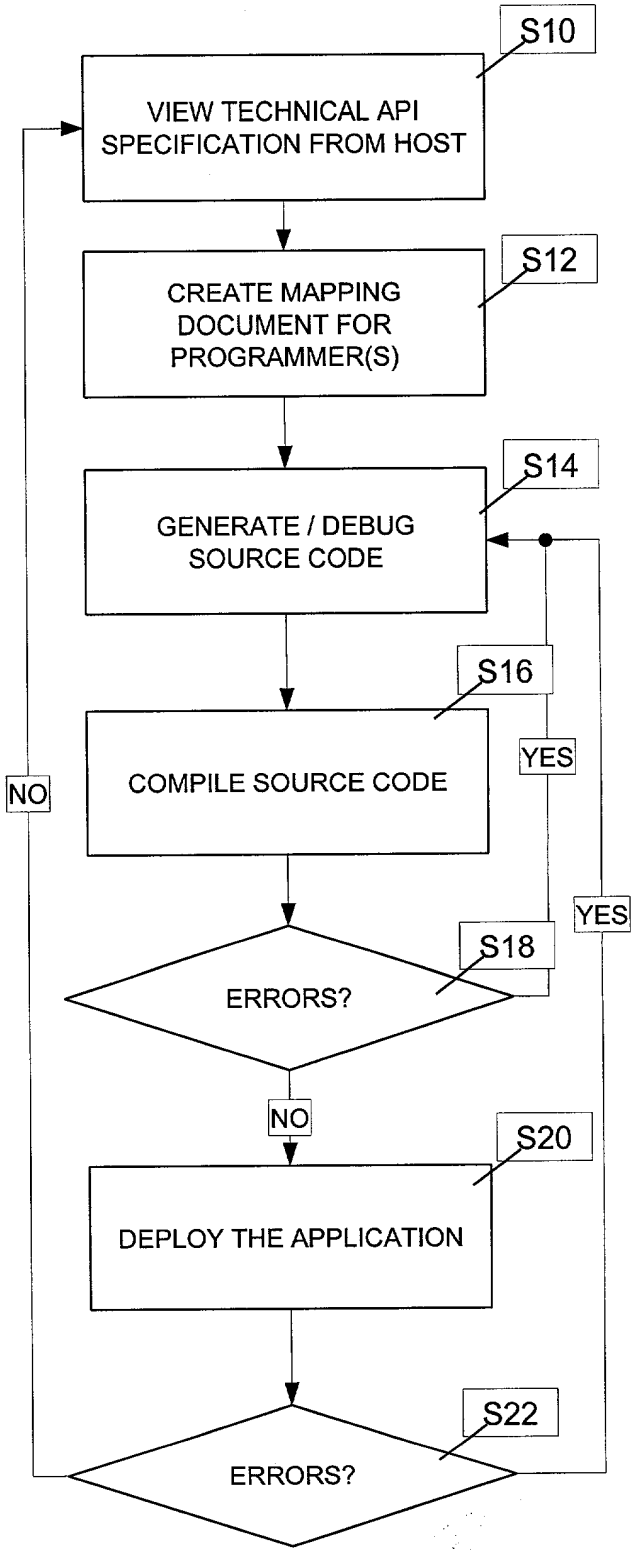
API Field Description	VT Server Field Definitions
Transaction Code	Hard code – set to '033'
Credit Union Access Code	Code retrieved from database configuration for each particular credit union
Member Number to Withdrawal Funds From	From internal field TransacAccntFrom before field separator.
Account Suffix of Withdrawal Funds From	From internal field TransacAccntFrom after field separator
Transfer Amount	From internal field TransacAmount. Internal field includes decimal point. External field does not. Remove decimal point before sending. Maximum amount is 9,999,999.99
Post Indicator	Set based on TransacPostMode. If 0, set to N. If 1, set to Y.

Response Packet

Field Description	Content
Transaction Code	Field is echoed. Not used on response.
Credit Union Access Code	Field is echoed. Not used on response.
Member Number	Field is echoed. Not used on response.
Host Response Code	External system's response code. Map to the VT Server response code based on the configuration tables.
Sign field	Positive/negative sign indicator for field that follows. Use to map appropriately.
Current Balance of Withdrawal From Account (before transfer)	Map to the ledger balance field of the internal message. QBT balance indicator for ledger is 1.
Sign field	Positive/negative sign indicator for field that follows. Use to map appropriately.
Available Balance of Withdrawal From Account (before transfer)	Map to the ledger balance field of the internal message. QBT balance indicator for ledger is 2.

PRIOR ART

Fig. 3



PRIOR ART

Fig. 4

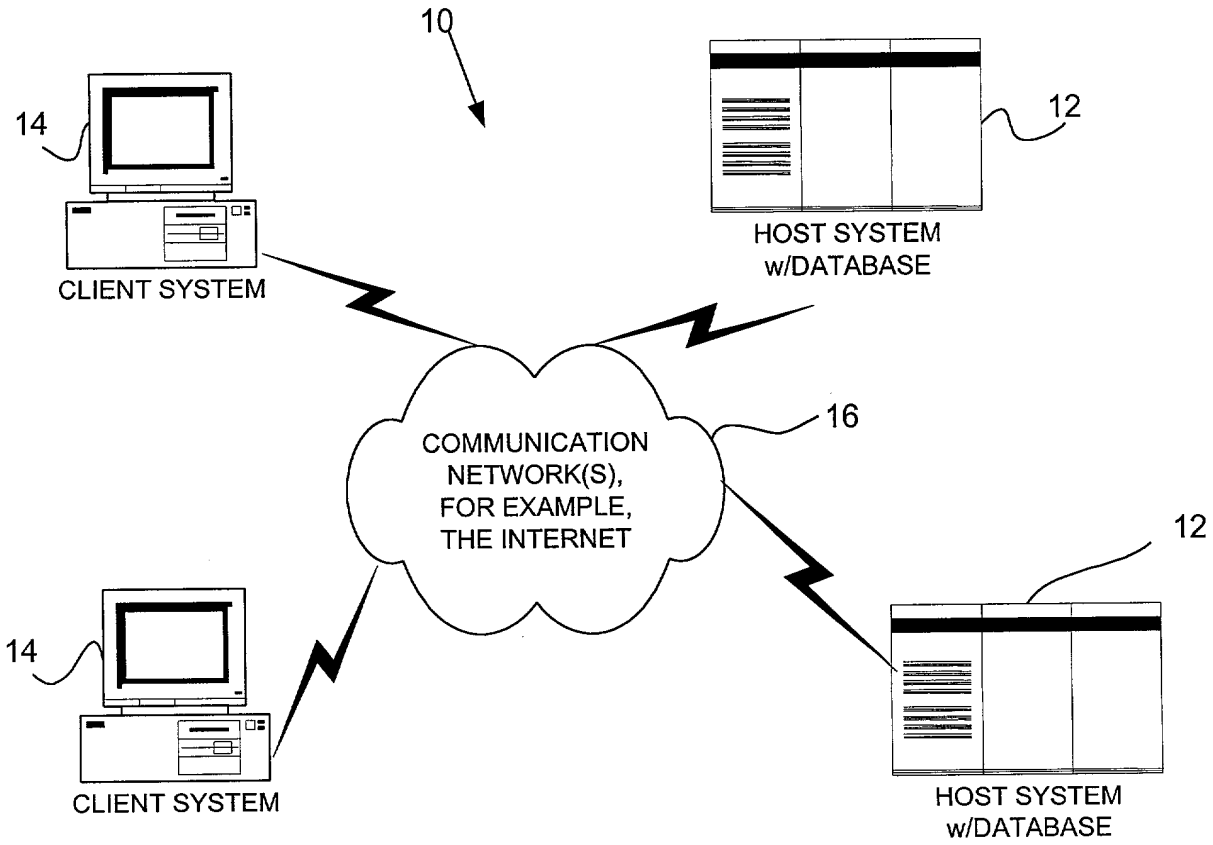

10
14
CLIENT SYSTEM
12
HOST SYSTEM
w/DATABASE
COMMUNICATION
NETWORK(S),
FOR EXAMPLE,
THE INTERNET
16
14
CLIENT SYSTEM
12
HOST SYSTEM
w/DATABASE

Fig. 5

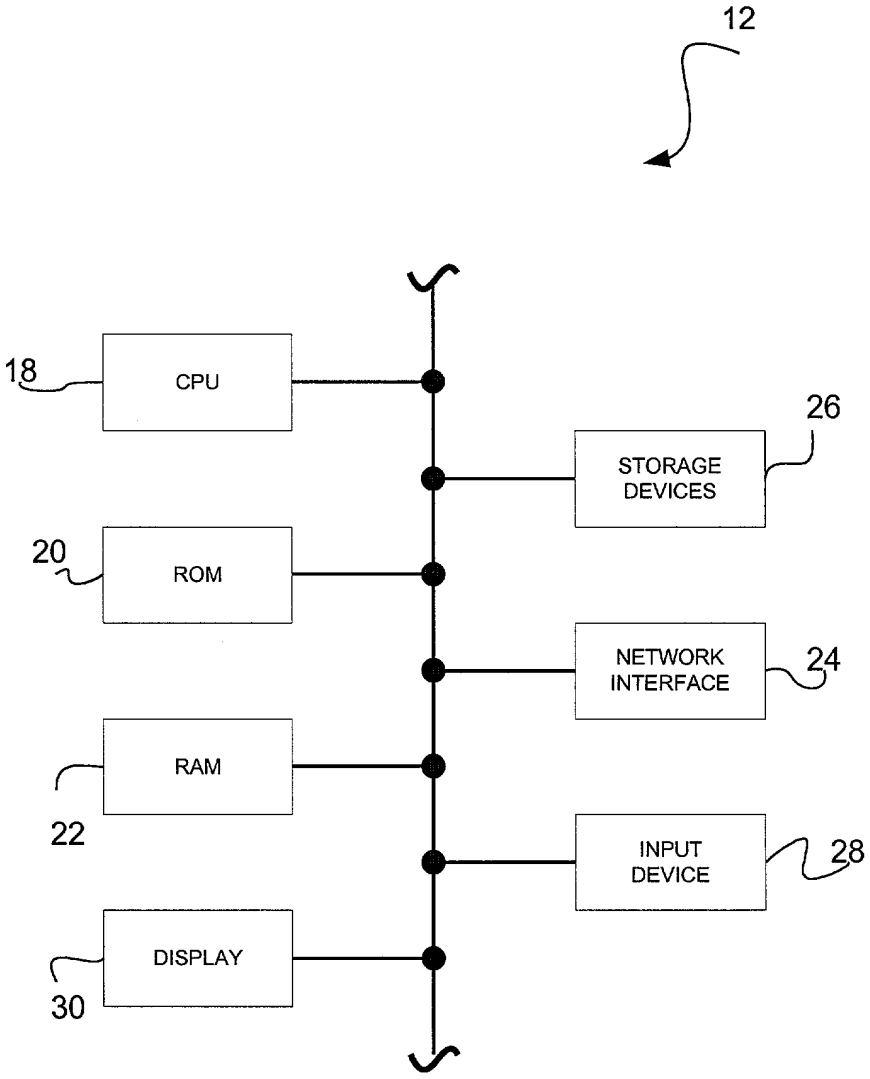

12
18
CPU
26
STORAGE DEVICES
20
ROM
NETWORK INTERFACE
24
RAM
22
INPUT DEVICE
28
DISPLAY
30

Fig. 6

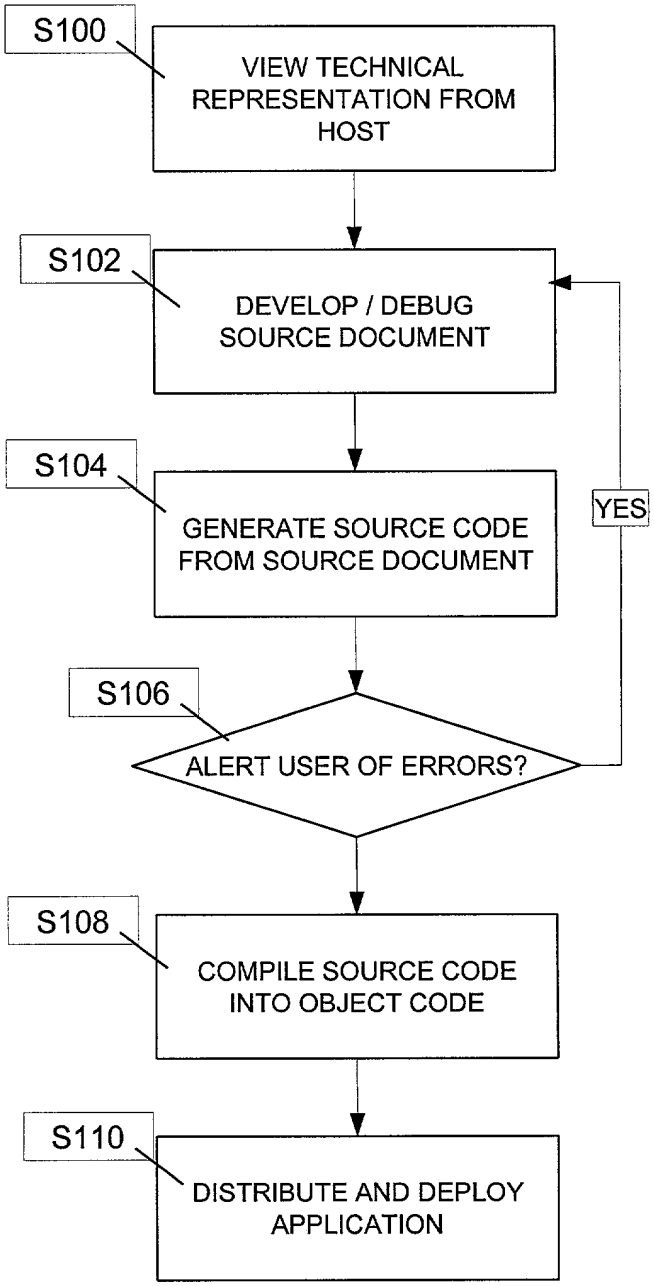

S100
VIEW TECHNICAL REPRESENTATION FROM HOST
S102
DEVELOP / DEBUG SOURCE DOCUMENT
S104
GENERATE SOURCE CODE FROM SOURCE DOCUMENT
YES
S106
ALERT USER OF ERRORS?
S108
COMPILE SOURCE CODE INTO OBJECT CODE
S110
DISTRIBUTE AND DEPLOY APPLICATION

Fig. 7

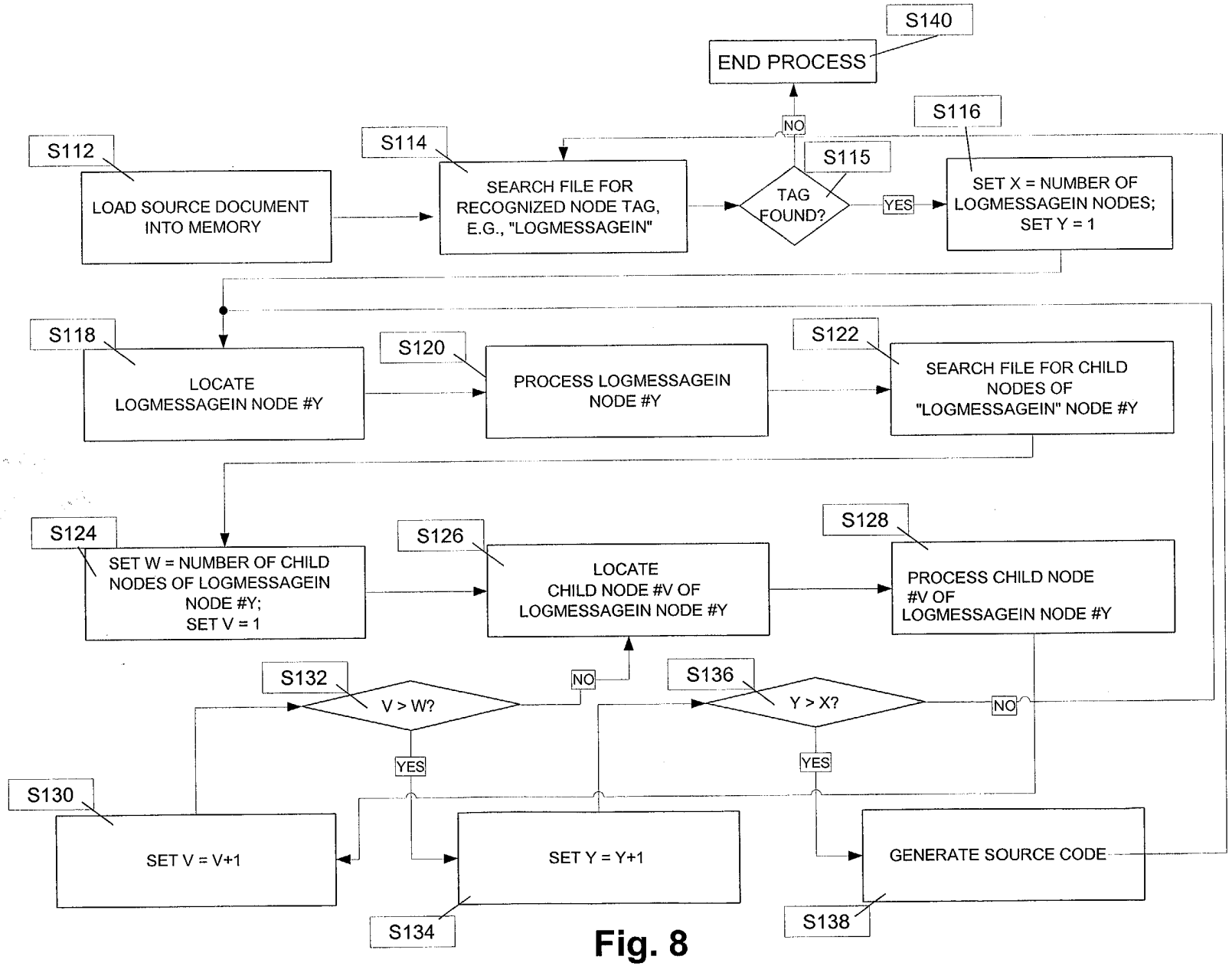


Fig. 8

```
1.      if ( !FAILED( spElement->getElementsByTagName(
2.              L"LogMessageIn", &spNodes ) )  && spNodes )
3.      {
4.              long lLength = 0;
5.              if ( FAILED( spNodes->get_length(&lLength)))
6.              {
7.                      return ;
8.              }
9.              CString sText;
10.             for ( long i = 0 ; i < lLength ;i++ )
11.             {
12.                     spNode.Release();
13.                     if ( !FAILED ( spNodes->get_item( i ,  &spNode ) ) )
14.                     {
15.                             CheckOutputLine( 0, spNode );
16.                             GetAttribute( spNode , L"id", sText );
17.                             if ( m_bDoVB )
18.                             {
19.                     sText = _T("Sub LogMessageIn_")+ sText + _T("( vtObj as VTMsgObj )");
20.                                     OutputLine( 0 , sText );
21.                                     OutputLine( 0, _T("\r\n") );
22.                                     OutputLine( 1, _T("'Temp variables used by routine\r\n"));
23.                                     OutputLine( 1, _T("Dim sTmp As String\r\n") );
24.                                     OutputLine( 1, _T("Dim sTmp4 As String\r\n") );
25.                                     OutputLine( 1, _T("Dim sTmp5 As String\r\n") );
26.                                     OutputLine( 1, _T("Dim sTmp3 As String\r\n") );
27.                                     OutputLine( 1, _T("Dim sTmp2 As String\r\n") );
28.                                     OutputLine( 1, _T("Dim sCmp As String\r\n") );
29.                                     OutputLine( 1, _T("Dim iOffset As integer\r\n") );
30.                                     OutputLine( 1, _T("\r\n"));
31.                                     OutputLine( 1, _T("On Error Goto ErrOut\r\n") );
32.                                     OutputLine( 1, _T("\r\n"));
33.                             } else
34.                             {
35.                                     AddCFunction( "LogMessageIn_" + sText);
36.                                     sText = _T("void MsgHandler::LogMessageIn_") + sText;
37.                                     sText += _T("( IDualVTMsgObj * vtObj )\r\n{\r\n\ttry {\r\n");
38.                                     OutputLine( 0 , sText );
39.                                     OutputLine( 0, _T("\r\n") );
40.                                     OutputLine( 1, _T("//Temp variables used byroutine\r\n"));
41.                                     OutputLine( 1, _T("CComBSTR sTmp;\r\n") );
42.                                     OutputLine( 1, _T("CComBSTR sTmp2;\r\n") );
43.                                     OutputLine( 1, _T("CComBSTR sTmp3;\r\n") );
44.                                     OutputLine( 1, _T("CComBSTR sTmp5;\r\n") );
45.                                     OutputLine( 1, _T("CComBSTR sTmp4;\r\n") );
46.                                     OutputLine( 1, _T("CComBSTR sCmp;\r\n") );
47.                                     OutputLine( 1, _T("int iOffset = 0, iLastPos = 0;\r\n") );
48.                                     OutputLine( 1, _T("\r\n"));
49.                                     OutputLine( 1, _T("//End of temp variables\r\n"));
50.                                     OutputLine( 1, _T("\r\n"));
51.                             }

52.                             m_bProcessingGetResponseCode += 1;
53.                             ProcessMessageIn( 1, spNode );
54                              m_bProcessingGetResponseCode  -= 1;

55.                             if ( m_bDoVB )
56.                             {
57.                                     OutputLine(0, _T("ErrOut:\r\n"));
58.                             OutputLine(1, _T("vtObj.EndSetErro Err.Number, Err.Description\r\n"));
59.                             }
```

Fig. 9

```
1. void CXMLEditDoc::ProcessMessageIn( int iTabIndex, CComPtr<IXMLDOMNode>&
spParentNode, BOOL bInBuildField, BOOL *bIfWasProccessed )
2.{
3.      CComPtr<IXMLDOMNode> spChild;
4.      if ( FAILED( spParentNode->get_firstChild( &spChild)) || !spChild )
5.      {
6.              retu rn;
7.      }

8.      CComPtr<IXMLDOMNodeList> spList = NULL;
9.      if ( !FAILED( spParentNode->get_childNodes( &spList ) )  && spList )
10.     {
11.             CComBSTR sNodeName;
12.             CComBSTR sText;
13.             CComPtr<IXMLDOMNode> spNode;
14.             long lLength = 0;
15.             spList->get_lengt  h( &lLength );
16.             for ( long i = 0 ; i < lLength ; i++ )
17.             {
18.                     spNode.Release();
19.                     sText.Empty();
20.                     sNodeName.Empty();
21.                     if ( !FAILED ( spList->get_item( i ,  &spNode ) ) )
22.                     {
23.                             spNode->get_nodeName( &sNodeName );

24.                             Che ckOutputLine( iTabIndex, spNode );

25.                             void *ptr = NULL;

26.                             CString strNodeName = sNodeName;

27.                             strNodeName.MakeUpper();

28.                             // now the ugly look up table
29.                             glb_MapOfXMLStringsTolds.Lookup( strNodeName, ptr );

30.                             int id = (int)   ptr;

31.                             switch ( id )
32.                             {
*
*
56.                             case IDTAG_BitmapDateIn:
57.                             {
58.                                     CString sBitPos;
59.                                     CString sLen;
60.                                     CString sTranField;
*
*                       {
74.                                                                     OutputLine( iTabIndex, _T("If
vtObj.IsBitmapPositionSet( ") , _T("if (IsBitmapPositionSet( vtObj, "));
                                                                        OutputLine( 0, sBitPos );
                                                                        OutputLine( 0, _T(") Then\r\n") , _T(")) {\r\n") );
                                                                        iTabIndex += 1;
                                                                        // first lets get the data
                                                                        CheckForPackedAttribute( iTabIndex, spNode );
```

Fig. 10

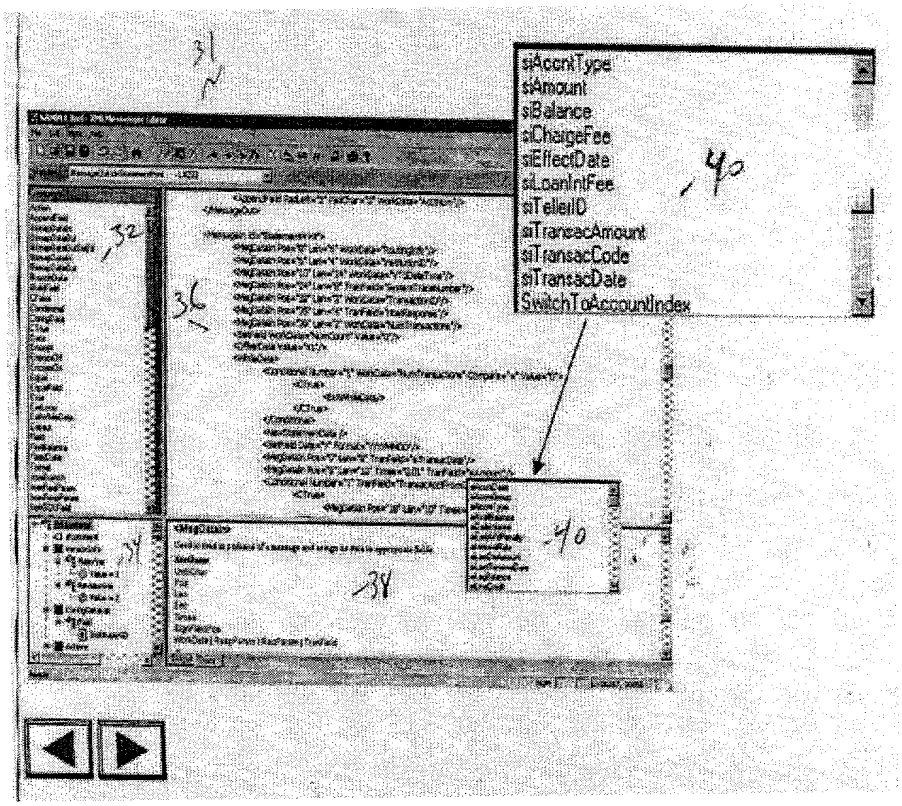


Fig. 11